The Gen-Pygments-CSS Documentation

Installation, usage instructions and API reference.

hreikin

Table of contents

1. The `gen-pygments-css` Docs

1.1 About The Project

Generate CSS stylesheets for each Pygments supported style.

- Github Link: https://github.com/hreikin/gen-pygments-css
- PyPi Link: https://pypi.org/project/gen-pygments-css/
- PDF Documentation: https://hreikin.github.io/gen-pygments-css/pdf/gen-pygments-css-documentation-LATEST.pdf

1.1.1 Built With

- Pygments
- Python

1.2 Installation

To get a local copy up and running choose one of the below install instructions and follow the steps provided.

1.2.1 Install With PIP

The simplest way to install `gen-pygments-css` is to use `pip`:

```
pip install gen-pygments-css
```

1.2.2 Install From Source

Alternatively you can install from source by following the steps below:

1. Clone the repo:

```
git clone https://github.com/hreikin/gen-pygments-css.git
cd gen-pygments-css/
```

2. Create and source a Python virtual environment:

```
python3 -m venv .venv
source .venv/bin/activate
```

3. Install requirements with `pip`:

```
pip install -r requirements.txt
```

1.3 Usage

CSS stylesheets for all `Pygments` styles are output into a created `css/` directory by default, this can be overridden.

```
from gen_pygments_css.gen_pygments_css import gen_pygments_css

# Called with no arguments.
gen_pygments_css()

# Called with a string passed into the styles_list.
gen_pygments_css(styles_list="monokai")

# Called with a list of strings passed into the styles_list.
gen_pygments_css(styles_list=["monokai", "stata-dark"])

# Call with a CSS selector defined.
```

```
gen_pygments_css(css_selector=".highlight")

# Call with a multiple arguments defined.
gen_pygments_css(css_selector=".highlight", styles_list=["monokai", "stata-dark"])

# Call with a relative custom css_dir.
gen_pygments_css(css_dir="assets/styles/")

# Call with an absolute custome css_dir.
gen_pygments_css(css_dir="/home/user/project/assets/styles/")

# Call the function and create a list of strings containing the paths of all
# stylesheets.
my_list = gen_pygments_css()
```

Last update: 2022-04-06

2. Reference Documentation

2.1 `gen_pygments_css.gen_pygments_css`

2.1.1

```
gen_pygments_css(css_dir='css/', styles_list=['default', 'emacs', 'friendly', 'friendly_grayscale',
'colorful', 'autumn', 'murphy', 'manni', 'material', 'monokai', 'perldoc', 'pastie', 'borland', 'trac',
'native', 'fruity', 'bw', 'vim', 'vs', 'tango', 'rrt', 'xcode', 'igor', 'paraiso-light', 'paraiso-dark',
'lovelace', 'algol', 'algol_nu', 'arduino', 'rainbow_dash', 'abap', 'solarized-dark', 'solarized-light',
'sas', 'stata', 'stata-light', 'stata-dark', 'inkpot', 'zenburn', 'gruvbox-dark', 'gruvbox-light',
'dracula', 'one-dark', 'lilypond'], css_selector=None)
```

Generate `CSS` stylesheets for each `Pygments` supported style.

CSS stylesheets for all `Pygments` styles are output into a created `css/` directory by default.

To generate a single stylesheet pass in a string containing the name of a `Pygments` style or to generate multilple stylesheets pass in a list of strings containing `Pygments` style names. More information is available at the `Pygments` documentation:

https://pygments.org/docs/api/?highlight=get_all_styles#pygments.styles.get_all_styles

A string, list or tuple containing the CSS selector(s) to prefix onto the generated `Pygments` classes can also be defined and passed in when calling the function. More information is available at the `Pygments` documentation:

https://pygments.org/docs/formatters/?highlight=get_style_defs#HtmlFormatter

Example Usage:

```
from gen_pygments_css.gen_pygments_css import gen_pygments_css

# Called with no arguments.
gen_pygments_css()

# Called with a string passed into the styles_list.
gen_pygments_css(styles_list="monokai")

# Called with a list of strings passed into the styles_list.
gen_pygments_css(styles_list=["monokai", "stata-dark"])

# Call with a CSS selector defined.
gen_pygments_css(css_selector=".highlight")

# Call with a multiple arguments defined.
gen_pygments_css(css_selector=".highlight", styles_list=["monokai", "stata-dark"])

# Call with a relative custom css_dir.
gen_pygments_css(css_dir="assets/styles/")

# Call with an absolute custome css_dir.
gen_pygments_css(css_dir="/home/user/project/assets/styles/")

# Call the function and create a list of strings containing the paths of all
# stylesheets.
my_list = gen_pygments_css()
```

Parameters:

Name	Type	Description	Default
`css_dir`	`str`	Relative or absolute string of the location to output the generated CSS stylesheets to.	`'css/'`
`styles_list`	`list(str`	A string or list of strings containing the name of `Pygments` styles. Defaults to a list of all style names returned by the `get_all_styles()` function from `Pygments`.	`['default', 'emacs', 'friendly', 'friendly_grayscale', 'colorful', 'autumn', 'murphy', 'manni', 'material', 'monokai', 'perldoc', 'pastie', 'borland', 'trac', 'native', 'fruity', 'bw', 'vim', 'vs', 'tango', 'rrt', 'xcode', 'igor', 'paraiso-light', 'paraiso-dark', 'lovelace', 'algol', 'algol_nu', 'arduino', 'rainbow_dash', 'abap', 'solarized-dark', 'solarized-light', 'sas', 'stata', 'stata-light', 'stata-dark', 'inkpot', 'zenburn', 'gruvbox-dark', 'gruvbox-light', 'dracula', 'one-dark', 'lilypond']`
`css_selector`	`Union[str, list, tuple]`	A string, list or tuple containing the CSS selector(s) to prefix onto the generated `Pygments` classes. Defaults to None.	`None`

Returns:

Type	Description
`file_list (list(str))`	A list of strings containing the location(s) of the generated CSS stylesheet(s).

t', 'solarized-light', 'sas', 'stata', 'stata-light', 'stata-dark', 'inkpot', 'zenburn', 'gruvbox-dark', 'gruvbox-light', 'dracula', 'one-dark', 'lilypond'], css_selector=None)

Source code in `gen_pygments_css/gen_pygments_css.py`

```
def gen_pygments_css(css_dir="css/", styles_list=list(get_all_styles()), css_selector=None):
    """Generate `CSS` stylesheets for each `Pygments` supported style.

    CSS stylesheets for all `Pygments` styles are output into a created `css/`
    directory by default.

    To generate a single stylesheet pass in a string containing the name of a
    `Pygments` style or to generate multilple stylesheets pass in a list of
    strings containing `Pygments` style names. More information is available at
    the `Pygments` documentation:

    <https://pygments.org/docs/api/?highlight=get_all_styles#pygments.styles.get_all_styles>

    A string, list or tuple containing the CSS selector(s) to prefix onto the
    generated `Pygments` classes can also be defined and passed in when calling
    the function. More information is available at the `Pygments` documentation:

    <https://pygments.org/docs/formatters/?highlight=get_style_defs#HtmlFormatter>

    Example Usage:

        from gen_pygments_css.gen_pygments_css import gen_pygments_css

        # Called with no arguments.
        gen_pygments_css()

        # Called with a string passed into the styles_list.
        gen_pygments_css(styles_list="monokai")

        # Called with a list of strings passed into the styles_list.
        gen_pygments_css(styles_list=["monokai", "stata-dark"])

        # Call with a CSS selector defined.
        gen_pygments_css(css_selector=".highlight")

        # Call with a multiple arguments defined.
        gen_pygments_css(css_selector=".highlight", styles_list=["monokai", "stata-dark"])

        # Call with a relative custom css_dir.
        gen_pygments_css(css_dir="assets/styles/")

        # Call with an absolute custome css_dir.
        gen_pygments_css(css_dir="/home/user/project/assets/styles/")

        # Call the function and create a list of strings containing the paths of all
        # stylesheets.
        my_list = gen_pygments_css()

    Args:
        css_dir (str): Relative or absolute string of the location to output the
            generated CSS stylesheets to.

        styles_list (list(str)): A string or list of strings containing the name
            of `Pygments` styles. Defaults to a list of all style names returned
            by the `get_all_styles()` function from `Pygments`.

        css_selector (Union[str, list, tuple]): A string, list or tuple
            containing the CSS selector(s) to prefix onto the generated
            `Pygments` classes. Defaults to None.

    Returns:
        file_list (list(str)): A list of strings containing the location(s) of
            the generated CSS stylesheet(s).
    """
    if isinstance(styles_list, str):
        styles_str = styles_list
        styles_list = list()
        styles_list.append(styles_str)
    css_dir = Path(css_dir).resolve()
    logging.info(f"Creating '{css_dir}' directory if it doesn't already exist.'")
    css_dir.mkdir(parents=True, exist_ok=True)
    file_list = list()
    logging.info("Generating CSS stylesheets.")
    for item in styles_list:
        file_location = Path(f"{css_dir.resolve()}/{item}.css")
        file_list.append(file_location)
        logging.info(f"Style: {item}")
        logging.info(f"Location: {file_location}")
        if css_selector == None:
            cmd_str = f"pygmentize -S {item} -f html > {file_location}"
            subprocess.run(cmd_str, shell=True)
        else:
            cmd_str = f"pygmentize -S {item} -f html -a {css_selector} > {file_location}"
            subprocess.run(cmd_str, shell=True)
    logging.info(f"Generated CSS stylesheets available at '{css_dir}'.")
    return file_list
```

Last update: 2022-04-06